UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/667,940	09/22/2003	Kyle R. Johns	MS1-1671US	1637

22801 7590 07/25/2005

LEE & HAYES PLLC
421 W RIVERSIDE AVENUE SUITE 500
SPOKANE, WA 99201

EXAMINER	
HSU, JONI	

ART UNIT	PAPER NUMBER
2671	

DATE MAILED: 07/25/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/667,940	JOHNS ET AL.	
	Examiner	Art Unit	
	Joni Hsu	2671	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☐ Responsive to communication(s) filed on _____.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-20 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-20 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date 10/6/04, 8/20/04.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Information Disclosure Statement

1. The information disclosure statements (IDS) submitted on October 6, 2004 and August 20, 2004 were filed after the mailing date of the application on September 22,2003. The submission is in compliance with the provisions of 37 CFR 1.97. Accordingly, the information disclosure statement is being considered by the examiner.

Specification

2. The disclosure is objected to because of the following informalities: According to MPEP § 608.01 (m), the present Office practice is to insist that each claim must be the object of a sentence starting with "I (or we) claim," "The invention claimed is" (or the equivalent). The phrase "Claims" is not considered equivalent to these appropriate phrases. Appropriate correction is required.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the

international application designated the United States and was published under Article 21(2) of such treaty in the English language.

4. Claims 1, 3-5, 7, 15, 17, and 19 are rejected under 35 U.S.C. 102(e) as being anticipated by Schenk (US 20040003370A1).

5. With regard to Claim 1, Schenk describes identifying a set of commands to be submitted to a processing unit (*associated code that can be executed to render the art asset*, [0087], *render method data elements transferred to the GPU*, [0082]); selecting a subset of the set of commands (*pointer to a byte code stream that must be executed to render the model*, [0088]); submitting the subset of the set of commands to the processing unit for processing (*render method data elements transferred to the GPU*, [0082]); and analyzing processing performed by the processing unit in response to the subset of the set of commands (*performing data transfer optimizations...simulating the contents of the GPU memory over the executing of the rendering of a model*, [0115]).

6. With regard to Claim 3, Schenk describes that the analyzing comprises showing how a scene would be drawn without texture or light, with textures and lighting, and skinned with textures and lighting [0120] (Figures 3a, 3b, 3c). Therefore, Schenk discloses showing how a scene would be drawn using only the subset of the set of commands.

7. With regard to Claim 4, Schenk describes that the processing unit comprises a graphic processing unit (120, Figure 6) [0046], and wherein the set of commands comprises commands

to be submitted to the graphics processing unit to have a frame drawn (*render method data elements transferred to the GPU*, [0082], *various elements of geometric data are coupled with a shading program at runtime to draw the asset*, [0020]).

8. With regard to Claim 5, Schenk describes performing data transfer optimizations by removing redundant transfers of code data to the GPU [0115]. Removing redundant transfers of code data inherently involved removing redundant transfers of data from the code that was previously submitted to the processing unit. Therefore, the set of commands must inherently be captured as they were previously submitted to the processing unit.

9. With regard to Claim 7, Schenk describes removing redundant transfers of code data to the GPU [0115]. Therefore, Schenk describes modifying one or more of the subset of the set of commands prior to submitting the subset of the set of commands to the processing unit.

10. With regard to Claim 15, Schenk describes one or more computer readable media [0128] having one or more instructions that, when executed by one or more processors, causes the one or more processors to capture a state of a graphics processing unit (*model is a collection of state objects (rendering states)*, [0054], *render method data elements transferred to the GPU*, [0082]); capture a plurality of commands submitted to the graphics processing unit in order to draw a frame of video (*various elements of geometric data are coupled with a shading program to draw the asset*, [0020], *geometric data are converted into data structure, and a code stream is assembled that describes the manipulations required to render these data structures*, [0021]);

and save the captured plurality of commands (*code is stored*, [0128]). Schenk describes that to capture the state of the graphics processing unit (120, Figure 6) is to obtain the state variables (*the front end provides the back end with a list of packets, each of which has an associated render method and set of input data. The input data is converted into the data defined in the variables section of the render method associated with the packet. The result is an instantiated packet. In an instantiated packet, the data for every variable is either known, or an external symbolic reference is known that will resolve to the memory location of that data at run time*, [0101, 0071]). The variables are registered [0067], meaning that the captured states are saved.

11. With regard to Claim 17, Schenk describes that to capture the state of the graphics processing unit (120, Figure 6) is to obtain the state variables (*the front end provides the back end with a list of packets, each of which has an associated render method and set of input data. The input data is converted into the data defined in the variables section of the render method associated with the packet. The result is an instantiated packet. In an instantiated packet, the data for every variable is either known, or an external symbolic reference is known that will resolve to the memory location of that data at run time*, [0101, 0071]). The variables are registered [0067], meaning that the variables are the settings of the registers. Therefore, to capture the state of the graphics processing unit is to obtain the settings of all registers of the graphics processing unit.

12. With regard to Claim 19, Schenk describes identifying a memory location reference by one of the variables; and capturing the contents of the memory location (*the data for every*

variable is either known, or an external symbolic reference is known that will resolve to the memory location of that data at run time, [0101]). Drawing a model is done by setting the required control variables [0054]. Therefore, the variables are considered to be the same as commands. Therefore, to capture the plurality of commands is to identify a memory location referenced by one of the plurality of commands; and capture the contents of the memory location.

13. Thus, it reasonably appears that Schenk describes or discloses every element of Claims 1, 3-5, 7, 15, 17, and 19 and therefore anticipates the claim subject.

Claim Rejections - 35 USC § 103

14. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

15. The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.

4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

16. Claims 2, 9-12, and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schenk (US 20040003370A1) in view of Thelen (US006557167B1).

17. With regard to Claim 2, Schenk is relied upon for the teachings as discussed above relative to Claim 1.

However, Schenk does not teach that the analyzing comprises measuring an amount of time taken for the subset of the set of commands to be processed. However, Thelen describes a computer program performance analyzer (131, Figure 1) that is a tool that is used to gather performance data for the computer program (125) to allow a user to identify performance bottlenecks and other pertinent characteristics of the computer program (Col. 3, lines 31-35). A code segment logger (133) logs data for each code segment as it runs. Once code segment logger has logged performance data for all code segments in the computer program, this logged performance data is used by a graphical program generator to generate a graphical representation of the computer program that includes the performance data. Once the graphical representation of the performance data is generated by the graphical program generator, a user may use the query tool to specify user-defined ad hoc queries on the performance data in the graphical representation. When the use submits a query, the computer program is "replayed" from the graphical representation, gathering information that satisfies the query at each step (Col. 3, line 53-Col. 4, line 5). The code segment logger logs the amount of time it took to execute the current code segment, and the other code segments that were called when the current code

segment ran (Col. 3, lines 54-57). Therefore, the analyzing comprises measuring an amount of time taken for the subset of the set of commands to be processed.

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to modify the device of Schenk so that the analyzing comprises measuring an amount of time taken for the subset of the set of commands to be processed as suggested by Thelen because Thelen suggests that this is needed in order to determine the performance bottlenecks that are cause by interactions between multiple code segments (Col. 1, lines 62-65).

18. With regard to Claim 9, Schenk describes one or more computer readable media [0128]. Schenk describes removing redundant transfers of code data to the GPU [0115]. Therefore, Schenk inherently discloses having one or more instructions that, when executed by one or more processors (*render method data elements transferred to the GPU*, [0082]), causes the one or more processors to identify a stream of commands previously submitted to a processing unit; modify the stream of commands (*removing redundant transfers of code data to the GPU*, [0115]); submit the modified stream of commands to the processing unit.

However, Schenk does not teach determining a difference between a first amount of time required by the processing unit to process the stream of commands and a second amount of time required by the processing unit to process the modified stream of commands. However, Thelen describes calling and executing different sequences of code segments (Col. 6, lines 6-23). Therefore, Thelen describes modifying the stream of commands and submitting the modified stream of commands to the processing unit. Thelen describes determining the amount of time required by the processing unit to process the different streams of commands (Col. 6, lines 31-

60), and the user analyzes the results (*gather and allow analysis of the performance data*, Col. 6, lines 6-8) to determine performance bottlenecks (*computer program performance analyzer 131 is a tool that is used to gather performance data for the computer program 125 to allow a user to identify performance bottlenecks and other pertinent characteristics of the computer program*, Col. 3, lines 31-35). Therefore, the user inherently determines a difference between a first amount of time required by the processing unit to process the stream of commands and a second amount of time required by the processing unit to process the modified stream of commands.

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to modify the device of Schenk to include determining a difference between a first amount of time required by the processing unit to process the stream of commands and a second amount of time required by the processing unit to process the modified stream of commands as suggested by Thelen because Thelen suggests that in order to determine the performance bottlenecks that are cause by interactions between multiple code segments (Col. 1, lines 62-65), the difference between the first amount of time and the second amount of time must be determined (Col. 5, lines 1-24).

19. With regard to Claim 10, Schenk describes that the processing unit comprises a graphics processing unit (120, Figure 6) [0046]. Schenk describes performing data transfer optimizations by removing redundant transfers of code data to the GPU [0115]. Removing redundant transfers of code data inherently involved removing redundant transfers of data from the code that was previously submitted to the processing unit. Therefore, the stream of commands must inherently comprise commands previously submitted to the graphics processing unit. The steam of

commands comprises commands to be submitted to the graphics processing unit to have a frame of video drawn (*render method data elements transferred to the GPU*, [0082], *various elements of geometric data are coupled with a shading program at runtime to draw the asset*, [0020]).

However, Schenk does not teach determining the first amount of time required by the processing unit to process the stream of commands and the second amount of time required by the processing unit to process the modified stream of commands. However, Thelen describes calling and executing different sequences of code segments (Col. 6, lines 6-23). Therefore, Thelen describes modifying the stream of commands and submitting the modified stream of commands to the processing unit. Thelen describes determining the amount of time required by the processing unit to process the different streams of commands (Col. 6, lines 31-60). The user can modify the stream of commands and determine the amount of time required by the processing unit to process the stream of commands and the modified stream of commands (Col. 6, lines 31-60). Therefore, Thelen describes determining the first amount of time required by the processing unit to process the stream of commands and the second amount of time required by the processing unit to process the modified stream of commands. This would be obvious for the same reasons given in the rejection for Claim 9.

20. With regard to Claim 11, Schenk describes that to modify the stream of commands is to remove one or more redundant commands (*removing redundant transfers of code data to the GPU*, [0115]).

21. With regard to Claim 12, Schenk describes that not all variables used by a shader or GPU can be made available through a compiler preprocess. For example, data such as transformation matrices are not available at compile time. They could be communicated as implicit variables. This would restrict the user from extending the set of such variables. The render method specification is extended through external linkage. By adding the extern keyword to a variable declaration along with an assignment of the form=variable_name, a reference is made to an external variable so that the internal variable is revealed [0067]. According to the disclosure of this application, an internal variable is a variable within the GPU (page 26, lines 6-9). Therefore, Schenk describes that to modify the stream of commands is to change one or more instructions of an internal program of the processor to reveal a value of an internal variable of the internal program.

22. With regard to Claim 18, Schenk describes drawing portions of the frame of video by executing code (*render method data elements transferred to the GPU*, [0082], *various elements of geometric data are coupled with a shading program at runtime to draw the asset*, [0020]).

However, Schenk does not teach that the one or more instructions further cause the one or more processors to capture timing data regarding how fast portions of the frame of video are drawn. However, Thelen describes that the one or more instructions further cause the one or more processors to capture timing data regarding how fast the code segment is executed (*code segment logger 133 logs the amount of time it took to execute the current code segment*, Col. 3, lines 54-57). This would be obvious for the same reasons given in the rejection for Claim 9.

23. Claims 6 and 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schenk (US 20040003370A1) in view of Thelen (US006557167B1), further in view of Levine (US005970439A).

24. With regard to Claim 6, Schenk and Thelen are relied upon for the teachings as discussed above relative to Claim 5.

However, Schenk and Thelen do not teach setting the processing unit, prior to submitting the subset of the set of commands to the processing unit, to a particular state, wherein the particular state is a same state as the processing unit was in at the time capture of the set of commands began. However, Levine describes setting the processing unit (10, Figure 1) to a particular state (*branch unit 20 inputs instructions and signals indicating a present state of processor 10*, Col. 5, lines 49-51), then submitting the subset of the set of commands to the processing unit (*in response to such instructions and signals, branch unit outputs signals to sequencer unit 18 signals indicating suitable memory addresses storing a sequence of instructions for execution by processor 10*, Col. 5, lines 51-54). The particular state is a present state of the processor. The state of various execution units are saved. This state of various execution units at the time the interrupt is signaled is provided in a saved state register. Thus, when the interrupt is actually serviced, the content of these registers provide the information concerning current instructions that are currently executing in the processor at the time of the signaling (Col. 9, line 66-Col. 10, line 6). Therefore, the particular state is a same state as the processing unit was in at the time capture of the set of commands began.

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to modify the devices of Schenk and Thelen so that setting the processing unit, prior to submitting the subset of the set of commands to the processing unit, to a particular state, wherein the particular state is a same state as the processing unit was in at the time capture of the set of commands began as suggested by Levine because Levine suggests that this is needed in order to debug and analyze the system by determining a machine's state at a particular point in time (Col. 2, lines 40-44), and making sure that the processing system captures the appropriate state (Col. 3, lines 7-29).

25. With regard to Claim 13, Schenk describes performing data transfer optimizations by removing redundant transfers of code data to the GPU [0115]. Removing redundant transfers of code data inherently involved removing redundant transfers of data from the code that was previously submitted to the processing unit. Removing redundant transfers of code data means that the stream of commands are modified. Therefore, the stream of commands were inherently captured as they were previously submitted to the processing unit, and the stream of commands are modified and submitted to the processing unit.

However, Schenk does not teach that the instructions further cause the one or more processors to set the processing unit, prior to submission of the modified stream of commands to the processing unit, to a particular state, wherein the particular state is a same state as the processing unit was in at the time capture of the stream of commands began. However, Levine describes setting the processing unit (10, Figure 1) to a particular state (*branch unit 20 inputs instructions and signals indicating a present state of processor 10*, Col. 5, lines 49-51), then

submitting the subset of the set of commands to the processing unit (*in response to such instructions and signals, branch unit outputs signals to sequencer unit 18 signals indicating suitable memory addresses storing a sequence of instructions for execution by processor 10*, Col. 5, lines 51-54). The particular state is a present state of the processor. The state of various execution units are saved. This state of various execution units at the time the interrupt is signaled is provided in a saved state register. Thus, when the interrupt is actually serviced, the content of these registers provide the information concerning current instructions that are currently executing in the processor at the time of the signaling (Col. 9, line 66-Col. 10, line 6). Therefore, the instructions further cause the one or more processors to set the processing unit, prior to submission of the stream of commands to the processing unit, to a particular state, wherein the particular state is a same state as the processing unit was in at the time capture of the stream of commands began. This would be obvious for the same reasons given in the rejection for Claim 6.

26. Claims 8 and 14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schenk (US 20040003370A1) in view of Thelen (US006557167B1) and Davidson (US006446029B1).

27. With regard to Claim 8, Schenk is relied upon for the teachings as discussed above relative to Claim 1. Schenk describes that the processing unit comprises a graphic processing unit (120, Figure 6) [0046]. Schenk describes performing data transfer optimizations by removing redundant transfers of code data to the GPU [0115]. This inherently involves analyzing the set of commands; determining, based on the analysis of the set of commands,

whether one or more recommendations for using the graphics processing unit are violated by the set of commands, the violation being redundant code. Schenk describes drawing portions of the frame of video by executing code (*render method data elements transferred to the GPU*, [0082], *various elements of geometric data are coupled with a shading program at runtime to draw the asset*, [0020]).

However, Schenk does not teach that if one or more recommendations are violated by the set of commands, then selection one of the violated recommendations; determining how much faster the frame could have been drawn if the selected recommendation had not been violated; and issuing a warning identifying both the selected recommendation that has been violated and how much faster the frame could have been drawn if the selected recommendation had not been violated. However, Thelen describes the user can analyze the set of commands by specifying queries on the performance data, and the computer program is "replayed" by stepping through the graphical representation, gathering information that satisfies the query at each step (Col. 3, line 63-Col. 4, line 5). Thelen describes determining, based on the analysis of the set of commands, whether one or more recommendations for using the processing unit are violated by the set of commands (*identify performance bottlenecks by defining program execution conditions 132 and running computer program 125 under those defined conditions*, Col. 3, lines 31-38). Thelen describes identifying performance bottlenecks by measuring the execution times of code segments so that the user can inherently compare the execution times and determine how much faster one code segment is than another (Col. 5, lines 1-23). Therefore, Thelen inherently discloses that if one or more recommendations are violated by the set of commands, then selecting one of the violated recommendations; determining how much faster the frame could

have been drawn if the selected recommendation had not be violated. This would be obvious for the same reasons given in the rejection for Claim 9.

However, Schenk and Thelen do not teach issuing a warning identifying both the selected recommendation that has been violated and how much faster the frame could have been drawn if the selected recommendation had not been violated. However, Davidson describes that if an instruction pipeline stage requires more time to complete than indicated by its corresponding threshold value, then thresholder (520, Figure 5B) asserts a threshold event signal 526 that is collected by an event counter 530 in the performance monitor (Col. 8, lines 62-66). The threshold event signal is a warning, and the threshold value is the selected recommendation. Therefore, Davidson describes issuing a warning identifying the selected recommendation that has been violated.

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to modify the devices of Schenk and Thelen to include issuing a warning identifying the selected recommendation that has been violated as suggested by Davidson because Davidson suggests the advantage of identifying the events that take longer than a threshold interval to execute so that these events can be collected and provided for optimization analysis (Col. 2, lines 36-56).

28. With regard to Claim 14, Schenk describes that the instructions further cause the one or more processors to analyze the stream of commands (optimization of code, [0114]). Schenk describes performing data transfer optimizations by removing redundant transfers of code data to the GPU [0115]. This inherently involves analyzing the set of commands; determining, based on

the analysis, whether one or more recommendations for using the processing unit are violated by the set of commands, the violation being redundant code; if one or more recommendations are violated by the stream of commands, then use, as the modified stream of commands, the stream of commands as modified to no longer violate a selected one of the one or more recommendations, in other words, using the code with the redundant code removed from it.

However, Schenk does not teach an indication of the difference. However, Thelen describes the user can analyze the set of commands by specifying queries on the performance data, and the computer program is "replayed" by stepping through the graphical representation, gathering information that satisfies the query at each step (Col. 3, line 63-Col. 4, line 5). Thelen describes determining, based on the analysis of the set of commands, whether one or more recommendations for using the processing unit are violated by the set of commands (*identify performance bottlenecks by defining program execution conditions 132 and running computer program 125 under those defined conditions*, Col. 3, lines 31-38). Thelen describes identifying performance bottlenecks by measuring the execution times of different sequences of code segments (Col. 5, lines 1-23) so that the user can inherently compare the difference between the execution times, and the user inherently uses the sequence of code segments that takes the shortest amount of time to execute (Col. 6, lines 31-60). This would be obvious for the same reasons given in the rejection for Claim 9.

However, Schenk and Thelen do not teach issuing a warning identifying both the selected recommendation that had been violated and an indication of the difference. Davidson describes that if an instruction pipeline stage requires more time to complete than indicated by its corresponding threshold value, then thresholder (520, Figure 5B) asserts a threshold event signal

526 that is collected by an event counter 530 in the performance monitor (Col. 8, lines 62-66). The threshold event signal is a warning, and the threshold value is the selected recommendation. Therefore, Davidson describes issuing a warning identifying the selected recommendation that has been violated. This would be obvious for the same reasons given in the rejection for Claim 8.

29. Claim 16 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schenk (US 20040003370A1) in view of Prindle (US 20030232648A1).

Schenk describes that the one or more instructions further cause the one or more processors to perform the captures and save (upload into memory, [0116]) in response to a request to capture the frame (CALL operations, [0107, 0117]).

However, Schenk does not teach that the request is received from a remote computing device. However, Prindle describes that the one or more instructions further cause the one or more processors (GPU 96, [0104]) to perform the captures and save in response to a request to capture the frame (*capturing video signals*, [0022]), wherein the request is received from a remote computing device (*captured video signal is sent from the CPU and passed to the requested device*, [0034], *client system 22, executable machine code 28 is read from a network server* [0061]).

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to modify the device of Schenk so that the request is received from a remote computing device as suggested by Prindle because Prindle suggests that this is needed in order to perform

real-time videoconferencing. For example, a first video game console can directly link with a second video game console [0021].

30. Claim 20 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schenk (US 20040003370A1) in view of Krueger (US006173368B1).

Schenk describes using memory locations that were referenced by a previous one of the plurality of commands (*the data for every variable is either known, or an external symbolic reference is known that will resolve to the memory location of that data at run time*, [0101], *dictionary data structure associated with the model that can be used at runtime to find exported variables. The symbolic names assigned to data in the prior pass are used to fill in pointers in the resulting dictionary,* [0104]).

However, Schenk does not teach determining whether the memory location was referenced by a previous one of the plurality of commands; if the memory location was not referenced by a previous one of the plurality of commands, then capture the contents of the memory location; and if the memory location was referenced by a previous one of the plurality of commands, then check whether the contents of the memory location are the same as the contents of the memory location when the memory location was referenced by the previous command, and capture the contents of the memory location only if the contents of the memory location are not the same as the contents of the memory location when the memory location was referenced by the previous command. However, Krueger describes that the instructions are retrieved from the cache (Col. 2, lines 45-49). The instructions are accessed by referencing memory locations (Col. 1, lines 54-57). Indicators are set in response to new data being received

by register 76. The new data is captured (Col. 16, lines 4-17). The data is inherently new if the memory location was not referenced by a previous one of the plurality of commands. Therefore, Krueger determines whether the memory location was referenced by a previous one of the plurality of commands; if the memory location was not referenced by a previous one of the plurality of commands, then capture the contents of the memory location. Krueger describes a C/D indicator (84) indicating whether the data is dirty or clean (Col. 14, lines 13-15). The data is dirty if it has been changed since it was read into register 76 (Col. 16, lines 19-26). If the data is dirty, then the data is captured to be written into the main memory (104, Figure 8; Col. 18, lines 47-54). The data is clean if the data in register 76 has been unchanged since it was read into register 76 (Col. 16, lines 19-22). If the data is clean, then it skips step 104, and therefore skips the capturing of the data (Col. 18, lines 35-41). Therefore, if the memory location was referenced by a previous one of the plurality of commands, then check whether the contents of the memory location are the same as the contents of the memory location when the memory location was referenced by the previous command, and capture the contents of the memory location only if the contents of the memory location are not the same as the contents of the memory location when the memory location was referenced by the previous command.

It would have been obvious to one of ordinary skill in this art at the time of invention by applicant to modify the device of Schenk to include determining whether the memory location was referenced by a previous one of the plurality of commands; if the memory location was not referenced by a previous one of the plurality of commands, then capture the contents of the memory location; and if the memory location was referenced by a previous one of the plurality of commands, then check whether the contents of the memory location are the same as the

contents of the memory location when the memory location was referenced by the previous command, and capture the contents of the memory location only if the contents of the memory location are not the same as the contents of the memory location when the memory location was referenced by the previous command as suggested by Krueger because Krueger suggests that data that has not changed does not need to be captured, and therefore this method minimizes accesses to the main memory (Col. 2, lines 2-7).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Joni Hsu whose telephone number is 571-272-7785. The examiner can normally be reached on M-F 8am-5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Ulka Chauhan can be reached on 571-272-7782. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

JH

Kee M. Tung
Primary Examiner